VOICE RELAY FOR DEAF USERS

by

Docas Zulu

A project report submitted in partial fulfillment of the requirements for the degree of

BSc Honours Computer Science

University of the Western Cape

2009

Date: November 4, 2009
Deaf people have restricted access to services when it comes to mobile phones as they cannot use the voice service of the phone. Communication is important for all users. This proposed system will make it easier for the hearing user to communicate with the Deaf user as the hearing person will be able to send a voice message as text to the deaf person. The system that will be designed will focus on the conversion service and the software will thus be on the server side, it will provide a communication bridge between a Deaf user and a hearing user, a hearing user will be able to call a Deaf user via a mobile phone and the Deaf user will receive a text message instead of voice and vice versa.

My project will implement the conversion of Text-to-Speech with Festival and Speech-to-Text using Sphinx. Asterisk will serve as a media gateway. The conversion will be achieved by making use of SIP (Session Initiation Protocol) on mobile phones. Voice Over IP will be used as transmission technology.
# TABLE OF CONTENTS

Abstract ................................................................................................................................. 2  
Table of Contents ................................................................................................................ 3  

**List of figures** .................................................................................................................. vii  
**List of Tables** .................................................................................................................. viii  
**Acknowledgments** ....................................................................................................... ix  
**Glossary** .......................................................................................................................... x  

**Chapter 1** ......................................................................................................................... 1  
   Introduction ......................................................................................................................... 1  
   Assisting the Deaf .............................................................................................................. 1  
   Literacy amongst the Deaf ............................................................................................... 2  
   Motivation for the research project ................................................................................. 2  
   Conclusion ......................................................................................................................... 2  

**Chapter 2** ......................................................................................................................... 3  
   User requirements document ......................................................................................... 3  
   User requirements gathering ........................................................................................ 3  
   The user’s view of the problem ..................................................................................... 3  
   Brief description of the problem domain ..................................................................... 4  
   Complete description of the problem domain ............................................................ 4  
   Software expectations ................................................................................................... 5  
   Not expected from the software ................................................................................... 6  

**Chapter 3** ......................................................................................................................... 7  
   Requirements analysis document ............................................................................... 7  
   Interpretation of the user’s requirements .................................................................. 7  
   Existing solutions ........................................................................................................ 8  
   Break the solution into parts ....................................................................................... 9  
   Communication ............................................................................................................ 9  
   Administrative Back-End .......................................................................................... 10  
   Dial plans ..................................................................................................................... 10  
   Ways to test the solution ............................................................................................ 11  

**Chapter 4** ......................................................................................................................... 13  
   User interface specification ....................................................................................... 13  
   Description of the complete user interface .............................................................. 13  
   How the interface looks like to the user .................................................................... 14  
   How the user interface behaves ............................................................................... 16  
   How the user interacts with the system .................................................................. 18  
   PJ SIP on PC (Deaf user) ......................................................................................... 18  
   Xlite softphone on PC (hearing user) ...................................................................... 18  
   Administrator ............................................................................................................ 18
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Manual for the hearing user and the Deaf user</td>
<td>63</td>
</tr>
<tr>
<td>Configuring X-lite</td>
<td>63</td>
</tr>
<tr>
<td>On screen Display</td>
<td>65</td>
</tr>
<tr>
<td><strong>Appendix E</strong></td>
<td>67</td>
</tr>
<tr>
<td>Questionnaires</td>
<td>67</td>
</tr>
<tr>
<td>Background questionnaires</td>
<td>67</td>
</tr>
<tr>
<td>System testing questionnaires</td>
<td>67</td>
</tr>
<tr>
<td><strong>Appendix F</strong></td>
<td>70</td>
</tr>
<tr>
<td>Project plan</td>
<td>70</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>1</td>
<td>Communication between Deaf and hearing users with a relay operator.</td>
</tr>
<tr>
<td>2</td>
<td>Communication between a hearing user and a Deaf user.</td>
</tr>
<tr>
<td>2</td>
<td>PJSIP on pc.</td>
</tr>
<tr>
<td>3</td>
<td>Xlite softphone settings for registration of accounts.</td>
</tr>
<tr>
<td>4</td>
<td>Xlite softphone used by hearing user for making and receiving calls.</td>
</tr>
<tr>
<td>5</td>
<td>CMD Line Interface of the Asterisk server.</td>
</tr>
<tr>
<td>6</td>
<td>PJSIP when sending an instant message.</td>
</tr>
<tr>
<td>7</td>
<td>shows the complete solution how the objects are connected.</td>
</tr>
<tr>
<td>8</td>
<td>usecase diagram illustrating the communication.</td>
</tr>
<tr>
<td>9</td>
<td>shows the one way communication for the Deaf User.</td>
</tr>
</tbody>
</table>
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Number</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1: User requirements</td>
<td>6</td>
</tr>
<tr>
<td>Table 2: Describes the identified objects in the solution</td>
<td>25</td>
</tr>
<tr>
<td>Table 3: illustrates the methods and attributes</td>
<td>37</td>
</tr>
</tbody>
</table>
ACKNOWLEDGMENTS

I would firstly like to thank my God for the strength that He gave me to do all this work because if it went for His mercy over my life and the grace to be patient and hardworking I don’t think I would have finished. Secondly I would like to thank my late mother Jean Rose Nkosi for her support and encouragement during my years at undergraduate level and most importantly my honours year. I will not forget my supervisor Dr William D. Tucker for his supervision and encouragement that I can do it and yes I did it. Lastly I would like to thank all my classmates because they all individually contributed a part to my honours year.
ASTERISK, is a software implementation of a telephone private branch exchange (PBX) capable of seamlessly bridging communication channels.

CMD, is a Windows Command File.

DCCT, a Community of Deaf people who help each other through self help. It was founded in 1987—a non government welfare organization whose aim is to address the needs of Deaf people in the Western Cape.

DEAF, refers to those who belong to the cultural community of Deaf people.

HLD, is a software approach that models a system as a group of interacting objects.

IP, A network protocol that specifies the format of data transferred between two hosts (called packets or datagrams) and the addressing scheme.

LLD, is the process of planning a system of interacting objects for the purpose of solving a software problem.

SIP, Session Initiation Protocol (SIP) is a signalling protocol, widely used for setting up and tearing down multimedia communication sessions video, chat, etc.

PBX, Private Branch Exchange is a switching system providing telephone communications between internal stations and external networks.

PJSIP, is an Open Source Embedded SIP protocol stack written in C.

PSTN, is a traditional phone system, using circuit switching to make and maintain connections for the duration of a phone call.

RAD, is a document that completely describes the system in terms of functional and nonfunctional requirements and serves as a contractual basis between the client and the developers.

URD, is a document used in software engineering that specifies the requirements the user expects from the software to be constructed in a software project.
**UIS**, is a model of the software that describes what the designed software looks like and how it behaves.

**VOIP**, voice over internet protocol means sending voice information in digital form in discrete packets rather than in the traditional circuit-committed protocols of the PSTN.
Communication is a way of interchanging thoughts, opinions or information by means of speech, writing or signs. Mobile technology enables communication. Not only are phones used for communications it can also be used for video conferencing and to play games. Some of these services that are provided by mobile network operators are not designed for the Deaf.

In the Deaf community there are Deaf people with severe auditory impairment and Deaf people that are prelingually deaf – these are people that are hearing impaired or have difficulties in hearing. This impairment implies that it is not easy for Deaf people to communicate with the hearing (Tucker, Glaser & Penton, 2002).

**Assisting the Deaf**

Literacy skills are essential to success in today’s technological society. Everyday examples of these skills include accessing the internet, sending and receiving emails; reading instructional manuals for the workplace etc. Therefore there are organizations around the world that assist the Deaf with skills acquisition. This includes improving their communication skills by improving their literacy. In Cape Town such an organization is the Deaf Community of Cape Town (DCCT)(Tucker,2008).
Literacy amongst the Deaf

Some of the Deaf at DCCT can write and read English. The few that can read and write have English skills that are very poor and learning to read and write is a very frustrating process to Deaf people because of their inadequate literacy experience in early childhood and delayed acquisition of vocabulary (Luckner, Sebald, Cooney, & Muir, 2006). Even if their English is poor they can still use SMS or Instant Messaging as form of communication. This means that they and hearing people can use SMS or Instant Messaging to communicate.

Motivation for the research project

The idea of this project is to design a communication bridge between the Deaf and hearing users. This means that a hearing user will be able to “talk” to the Deaf user and the Deaf user will receive a text message of the voice message.

Conclusion

In this chapter the background of the research topic was explained and some reference was to the literature available in the field. In the next chapter the user requirements will be discussed.
USER REQUIREMENTS DOCUMENT

The previous chapter described the problem at hand namely why Deaf and hearing users need a communication bridge. The User Requirements Document discusses the user's view of the problem, what the users require. It broadly describes the problem domain, the reason why this project has to be designed.

User requirements gathering

The honours project is based on the server side (the backend) and no new functionality will be added to the way instant messaging works, the existing way of sending an instant message, SMS and calling will be used in this honours project just that it will be the combination of the two. Therefore the user requirements where gathered from a Doctor of Philosophy thesis written by William D. Tucker, the thesis was used as a source of information for this honours project because it perfectly described the current problems that both Deaf and hearing users are facing when they want to communicate using telephones, it illustrated the need for an automated voice relay in the Deaf community

The user's view of the problem

The current communication between a Deaf user and a hearing user is through asynchronous communication. The problem that hearing users are facing is that when a user wants to make an urgent call, sending SMS or an instant message may be the choice but it is time consuming because of the time it takes to type
the message. The amount of time it takes to type will be less if one user sends SMS or instant message and the other replies with a voice message, thus increasing the communication speed.

**Brief description of the problem domain**

Communication using instant messaging/SMS and voice messaging between Deaf and hearing users is possible through the assistance of a relay operator. This provides a human communication bridge which does not solve the problem but slows down the communication speed because of the fact that the user has to request the service first.

**Complete description of the problem domain**

A request has to be made to the relay operator before communication between the Deaf and a hearing can begin. When the request has been granted the relay operator will then read aloud the message from the Deaf user to the hearing user and in reply the relay operator will type the voice message and send it to the Deaf user (See Figure 1)(Tucker, 2009). The problem with this relay operator is that only one user can call at a time and therefore time consuming, thus very inconvenient.

The honours project focuses on a combination of synchronous and asynchronous communication where Deaf users will be able to communicate with hearing users using either their mobile phones or instant messaging clients but without the assistance of a human relay operator.
Figure 1: Communication between Deaf and hearing users with a relay operator.

Software expectations

Hearing users should be able to make and receive calls whereas Deaf users should be able to send and receive text. The Deaf user will communicate using an Instant Messenger on a PC or a mobile phone and a hearing user will use either a landline phone, mobile phone or VOIP on a mobile phone (see table 1).

<table>
<thead>
<tr>
<th>Deaf user</th>
<th>Hearing user</th>
</tr>
</thead>
<tbody>
<tr>
<td>➢ Send/receive text</td>
<td>➢ Make and receive calls</td>
</tr>
<tr>
<td>➢ Instant Messaging</td>
<td>➢ IP phone or Cell phone</td>
</tr>
</tbody>
</table>

Table 1: User requirements
The system should be understandable and easy to use so that users would not find it difficult to register their new accounts. The user should be able to find help when stuck, troubleshoot a problem and precautions should be provided when necessary. The system should be user friendly, it should not be highly technical because it is developed for user satisfaction not developer satisfaction. A hearing user should be able to choose that which voice would he/she like to listen the new message with and this should not need telecommunication expect, there should be a default voice and also the option of using an alternative voice. The system should give the option of repeating a message in the hearing user's side and a Deaf user should be able to read a message more than once.

**Not expected from the software**
The system is only used as a calling and texting for hearing and Deaf users. It does not have video support. This means that the system is not for video calls but for voice calls and text messages only. The system is not expected to work for all languages it will be designed for English only, i.e. hearing users should not send voice messages using their mother tongue.

In the URD the requirements were stated and based on the requirements the behaviour of the system was identified. In the next chapter the requirements will be analyzed.
The requirements analysis document (RAD) is the interpretation of the user’s requirements from the designer’s point of view. This document focuses on what is needed so that the project may be successfully developed, how we go about implementing the user’s requirements. The RAD breaks user requirements analysis document (URD) into modules and defines their purpose and relationships.

**Interpretation of the user's requirements**

The system should provide a communication bridge between a Deaf user and a hearing user. The system required involves three users the Deaf user, Hearing user and the administrator and each of them have different requirements. The administrator will be controlling and maintaining the bridge between the Deaf and the hearing, the administrator will be the one giving users their contact details. For VoIP phones the users will be given username, password and domain name which is the IP address of the Asterisk server and extensions that they will use, whereas for non VoIP phones users will be given extensions, username and password.

For the Deaf user to contact someone, he/she must have a list of contacts info, therefore a Deaf user must be able to add and remove contact details for all users he/she would like to communicate with, and the contact list should be modifiable and viewable.
Hearing users will be calling from landline or mobile phones. Hearing users on mobile phones should be able to delete, add, modify and view contact information of all Deaf users, this will only be possible on the mobile phones not on landline phones because of their architecture.

**Existing solutions**

A fully automated relay system will be needed to replace the relay operator and for this to be possible, open source tools Sphinx and Festival will be used. Festival is a tool that will convert the text message to speech (The University of Edinburg, 2009), Festival is a text-to-speech system that converts normal language text into speech, there are voices that as part of the package the user will use for listening the messages with whereas Sphinx is a real-time speaker-independent speech recognition tool that will convert voice to text (Aksyonoff, 2009). Festival and Sphinx have been chosen to take part in the development process because both tools are open source tools and they both have been identified as the tools that work best with Asterisk.

The Asterisk has been identified as a potential media gateway for the system; it is a software “Private Branch Exchange” that has support in Automatic Speech Recognition and Text-to-Speech. The existing solution is better illustrated in the diagram below *(see figure 2)*. The idea of the automated relay is to integrate Festival and Sphinx into Asterisk, the integration will be achieved using the dial plans in Asterisk. Asterisk has to be configured such that when a text or voice message is received Asterisk will know what to do with the message before it is routed to the correct device, in other words it should know when to invoke Festival and when to invoke Sphinx, which means that when its text Festival has to be called and when its speech Sphinx has to be called. Asterisk server will transform each message that is received when applicable because there will be situations where the hearing user would text to the Deaf user where the invoking of the tools will
not be required. In addition, a hearing user should be able to choose which voice he/she will like to use for each and every contact person in the dial plan.

Figure 2: Communication between a hearing user and a Deaf user.

Break the solution into parts

Communication

Users who would like to use Voice over IP (VoIP) should have mobile phones that have Session Initiation Protocol (SIP) clients enabled in their phones, this means that the project will have to also use VoIP functionality to send SIP based communication to the Asterisk server. Asterisk is capable of providing PSTN breakout through the use of a PSTN interface card; this implies that a hearing user calling from a landline phone will be able to communicate with a Deaf user on a mobile phone, i.e. communication from SIP to PSTN (IP phone to landline phone). The system should allow a Deaf user to send text to the hearing user via Asterisk server, the hearing user will then “hear” the Deaf user’s typed text; the text string is converted to speech using Festival and when the hearing user replies the speech is converted to text using Sphinx.
Administrative Back-End

In addition, the system should have administrative control so as to register the user's SIP numbers to Asterisk, the administrator should be able to add and remove phones from the system according to their respective functionality (the VOIP mobile phones and the landline phones have distinctive requirements for registering the users). The administrator should be able to modify users accounts and view user accounts. When adding users on the system the administrator should be able to register Deaf users separately from hearing users, there should be a group of Deaf users and a group of hearing users. The administrative back-end will be an external system that manages the information needed by both the users and the Asterisk server so as to communicate. It would be used to modify the Asterisk’s configuration files to reflect and implement changes in the way the Administrator sets up the system. The administrative back-end will provide the connection (bridge) that is needed to enable the users to communicate.

Dial plans

Since there will be an integration of Sphinx and Festival, the Asterisk server will have dial plans that will give the extensions of each and every user that describe how Asterisk server is going to manage the Deaf and the hearing users when they are communicating. The dial plans will control how the order of calling will take place between users. So in the case of a hearing user there will be dial plans that will call Festival, in all users that will require Festival their extensions will have a way of invoking Festival. Deaf users are the ones that will require Sphinx so in all extensions of Deaf users Sphinx will be invoked; since the system will have a group of Deaf users with their accounts then in all users Sphinx will be invoked within Asterisk. The dial plans will also have extensions that direct voice calls to voice mail in case a user is absent.
There are configuration files that store the dial plans, this file consists of the extensions of the users and how asterisk will handle the procedure of calling. Configuration files are used by Asterisk to control its dial plans, i.e. the way Asterisk handles the calling procedure and access control when required. These files store all information that has to do with the user that may include user name, passwords.

**Ways to test the solution**

The system is an automated relay system that will benefit both the Deaf user and the hearing user. Therefore the system will be taken to a Deaf user and the hearing user, to observe the time complexity of the system because it should be faster than the use of a relay operator since there is no request of service. The system will be checked that the switching of the tools Festival and Sphinx happens as expectedly, meaning that the Deaf user receives text not voice and the hearing user receives voice not text. The voice messages to the hearing user will be checked to ensure that the user can hear the message clearly and can hear the right message from the right person, that the message destination is not mixed up.

The external system, administrative back-end will also be given to one of the users either Deaf or hearing, to test the friendliness of the system that the users are not finding any difficulties to find the appropriate window to add contacts, edit contacts and so on. Based on the feedback got from the users the system will be improved and tested again to ensure user satisfaction.

In the RAD the essential components needed to complete the system are identified and their relation to each other. In the next chapter, the description of the user interface will be discussed; its appearance to the user will be shown and explained.
USER INTERFACE SPECIFICATION

The user interface specification firstly identifies all the user interfaces, describes how the components work, what the interface is going to do and how the user will interact with the system.

Description of the complete user interface

There are Command Line(CMD) interfaces that will appear on the Deaf and interfaces that will be used by the hearing user. The first one is PJSIP client on PC which will be used by the Deaf user for instant messaging, i.e. when the user is sending text to the hearing user. Secondly is where by the Deaf users will register their accounts which will be contacts details of peers and will also contain connection information that will be needed by Asterisk server. There will also be another PJSIP CMD that will be for forwarding text messages from Asterisk to the Deaf user and from Deaf user to Asterisk, this PJSIP client will be used as a gateway between Asterisk and the PJSIP client on PC. The CMD of Asterisk which is what establish connection between the users will be used by the administrator to start the Asterisk server.

The hearing user will use a any sip softphone to send and receive voice messages. The interface for the hearing user is the sip settings account which shows how the user will register user details and connections to the Asterisk server. The interface below is for the hearing user on a softphone(figure 1), it shows how the users are going to register their accounts.
How the interface looks like to the user

Figure 2 below the PJSIP client on the PC, the instant messaging client used by the Deaf user running on Windows XP machine. This interface will appear after the user has compiled the PJSIP.

![PJSIP on PC](image)

*Figure 2  PJSIP on pc*

Figure 3 below shows how the user will register his/her account in order to establish connection to the Asterisk server, this interface will appear in the hearing user, it is the interface from the Xlite softphone on pc.
Figure 3 Xlite softphone settings for registration of accounts

The following Figure 4 below is the xlite softphone that will be used by the hearing user, on the screen is what will appear when the user is registered successful i.e. once asterisk server is started. This client will be used by the hearing user to make and receive calls.

Figure 4 Xlite softphone used by hearing user for making and receiving calls
The CMD of the Asterisk server (depicted by Figure 5 below) after it has been started by the administrator. The administrator will need to start the server before users will be registered, which will enable the users to communicate to one another.

![Image of CMD Line Interface of the Asterisk server]

**Figure 5 CMD Line Interface of the Asterisk server**

**How the user interface behaves**

Suppose a Deaf user wants to communicate with the hearing user, first of all the Deaf user and the hearing user needs to be registered to the Asterisk server. The Deaf user will type the hearing user's address and after that the Deaf user will type the desired message and then send the message to the hearing user. Figure 6 depicting what the interface does when the Deaf user sends an instant message, it shows the address of the message destination, the address of the sender which in our case the Deaf user and also the message content. After the message has been sent to the hearing user a notification message will be sent to the Deaf user.
informing the Deaf user whether the message has been successfully delivered or failed. It will appear on the same command line of PJSIP which is Figure 6 below.

![PJSIP output](image)

**Figure 6 PJSIP when sending an instant message**

Now when the hearing user is receiving the softphone will show if the user wants to reject the call or accept the call, it will show when the call is established and when call is ended. The softphone will then say the voice message from the Deaf user once the call has been established (depicted by Figure 7 below).
How the user interacts with the system

Asterisk only allows registered clients to communicate to one another, the following are steps needed to register a user to asterisk which is done by the Deaf user on the instant messenger PJSIP (from Figure 2). Then the steps that the hearing user must undergo to register (from Figure 3) and make call and receive calls from a Deaf user (from Figure 4).

PJSIP on PC (Deaf user)

- Choose +a to register
- User will specify the URL where PJSIP is running
- User will specify URL of Asterisk server
- User will specify username and password, which will correspond with the one in Asterisk
- Then the asterisk server will authenticate the user, once authenticated successfully the user will begin sending text
- The user will choose i to send an instant message
- User specify the username and the URL of Asterisk server
- Then type the message and the message will be forwarded to the hearing user

Xlite softphone on PC (hearing user)

The user will specify the accounts details of the user as shown at Figure 3 and after registration the registration has been authenticated by the Asterisk server the user will be able to receive and make calls.

Administrator

The administrator will also have an interface that will register the users, the administrator will provide both the Deaf and the hearing connection to Asterisk server.
Chapter 5

HIGH LEVEL DESIGN

This chapter is all about the objects identified by the RAD. The high level design gives an object oriented view of the problem, it provides a detailed description of each object of which are object that are needed as part of the solution.

Data dictionary

<table>
<thead>
<tr>
<th>Object</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PJSIP on PC</td>
<td>PJSIP is an instant messaging tool that can receive text and voice. It uses SIMPLE to send instant messages, it cannot store messages in case the user is offline it can also reject instant messages if the user is not registered on Asterisk server. It automatically registers itself to Asterisk server after 300 seconds. It is responsible for sending and receiving text to Asterisk server. The backend of the PJSIP will also be used to retrieve information from the database (user details) so as to identify the extension of the user before the message is routed to Asterisk server since the client on PJSIP on window will be identified by the name of the user.</td>
</tr>
<tr>
<td>SIP client on softphone(Xlite)</td>
<td>This is a softphone that uses SIP for routing calls, it will be responsible for</td>
</tr>
</tbody>
</table>
receiving and accepting voice calls for all users on Asterisk server. It can receive and send voice calls to users on its accounts. It can communicate with any user as long as both users are registered on the same server.

| Asterisk server | This is the main part of the solution; Asterisk server is a VoIP server software. This is where the integration of modules that are performing the transformation of voice and text messages happens, this server can accept, reject and redirect calls to the destination address. It supports SIP and IAX2 based communication. It can not only support voice calls but it is also capable of sending text to instant messaging clients within a call and SMS messages to mobile phones with an appropriate SMS gateway. |
| Festival server | Festival is a text-to-speech engine that deals with the transformation of text to voice, it can accept any text that is written in English and read what the text says. It is responsible for converting the text message to voice once it reaches Asterisk server, it does not route the message to the destination but only converts it. |
| Sphinx server | Sphinx is an automatic speech recognition (ASR) engine that accepts voice messages and transforms them into text; it can only understand voice messages in English. It is responsible for converting any voice message from the SIP client on softphone(Xlite). It does not send the message but only converts it. |

Table 2: Describes the identified objects in the solution
Class diagrams

The class diagram shows the whole solution of the problem, it shows how each object is related to another object, it shows the operations that each class is going to take. It illustrates how Asterisk server acts as the main part of the solution and how the other object mentioned in the data dictionary (Table 2).

Figure 7: shows the complete solution how the objects are connected

Relationships between the classes

The Asterisk server is the object that provides communication between the clients and the servers,
- PJSIPClientOnPC(Ubuntu) make use of the Asterisk server as the instant message gateway for the PJSIPClientOnPC i.e. when a instant message is sent to the SIPClientOnMobilePhone the message will go via the PJSIPClientOnPC(Ubuntu).
- PJSIPClientOnPC(windows) is associated to the PJSIPClientOnPC(Ubuntu) this two objects share similar attributes and some operations.
- AsteriskServer make use of FestivalServer and the SphinxServer for message transformation.
- SIPClientOnPC makes use of the AsteriskServer for establishing connection to the other user PJSIPClientOnPC(windows).
- The AdministrativeInterface allows the clients PJSIPClientOnPC(windows) and the register their account details and use the service.

![Usecase diagram illustrating the communication between users](Figure 8: usecase diagram illustrating the communication between users)
This chapter gives the pseudo code for the classes defined in the previous chapter, it gives the methods and attributes of the classes. It also explains what the attributes are for, what they do.

**Inner details of class attributes**

<table>
<thead>
<tr>
<th>Class</th>
<th>Attributes(data types)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PJSIPClientOnPC(windows)</td>
<td><strong>int addressURL</strong>: stores the IP address of the windows machine where PJSIP is installed.</td>
</tr>
<tr>
<td></td>
<td><strong>int time</strong>: stores the time an instant message was sent and received to and from the SIP client on the mobile phone.</td>
</tr>
<tr>
<td></td>
<td><strong>String username</strong>: stores the name of the user that is the same as the one in the Asterisk server.</td>
</tr>
<tr>
<td>String password: stores the password of the user that will be used when Asterisk server authenticates the user.</td>
<td></td>
</tr>
<tr>
<td>int registrarURL: stores the URL of Asterisk server used when registering the user in Asterisk server.</td>
<td></td>
</tr>
<tr>
<td>int portno: stores the port number of Asterisk server.</td>
<td></td>
</tr>
<tr>
<td>Int message: stores the content of the instant message</td>
<td></td>
</tr>
<tr>
<td>AdministrativeBackend</td>
<td></td>
</tr>
<tr>
<td>int extension: stores the extension that will identify the hearing user.</td>
<td></td>
</tr>
<tr>
<td>int username: stores the username of both SIP clients for registration to Asterisk server.</td>
<td></td>
</tr>
<tr>
<td>int password: stores the password to authenticate the SIP clients.</td>
<td></td>
</tr>
<tr>
<td>SIPClientOnMobilePhone</td>
<td></td>
</tr>
<tr>
<td>int username: stores the username as identified by the Asterisk server.</td>
<td></td>
</tr>
<tr>
<td>Class name</td>
<td>Methods(functions)</td>
</tr>
<tr>
<td>------------</td>
<td>--------------------</td>
</tr>
</tbody>
</table>
| PJSIPClientOnPC(windows) | **Void SendIM()**: this method it sends the instant message, it carries the content of the message once and send it to destination address.  
**Void ReceiveIM()**: this method enable the users to receive the content of the message.  
**Void addContacts()**: allows the user to add contact details of peers, it gets and sets the username and address of other SIP clients.  
**String to()**: takes the address of the destination user as input from the user. |
| **int domain**: stores the domain name of the Asterisk server. |  
**int password**: stores the password of the user as given by Asterisk server. |

**Inner details of methods**
<table>
<thead>
<tr>
<th><strong>PJSIPClientOnPC(Ubuntu)</strong></th>
<th><strong>String From()</strong>: carries the address of the sender.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>String AcceptIMfromAsterisk()</strong>: returns the message from Asterisk, which was sent by the SIP client on mobile phone.</td>
</tr>
<tr>
<td></td>
<td><strong>String SendIMtoAsterisk()</strong>: returns the message that was sent by the PJSIP client on PC(windows) to Asterisk server.</td>
</tr>
<tr>
<td></td>
<td><strong>Void connectToDatabase()</strong>: this method it establish connection to the database to get the extension of the destination user.</td>
</tr>
<tr>
<td></td>
<td><strong>int GetExtension()</strong>: it gets the extension of the destination user and return an int.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>int setExtension()</td>
<td>changes the extension from the old value to a new value for the current user who is requesting this service.</td>
</tr>
<tr>
<td>Void DisconnectDatabase()</td>
<td>it disconnects the user from access to data in the database.</td>
</tr>
<tr>
<td>String AcceptIMfromClient()</td>
<td>it returns the instant message content from the PJSIP client on PC(windows).</td>
</tr>
<tr>
<td>String RejectIMfromClient()</td>
<td>it returns the message from the user and return it back to the sender when user is offline.</td>
</tr>
<tr>
<td>AdministrativeBackend Void addHearingUser()</td>
<td>add the contact details of all the hearing users that will be using the system.</td>
</tr>
<tr>
<td>Void addDeafUser()</td>
<td>addDeafUser(): add the contact details of all Deaf users that will make use of the system.</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>String getHearingUser()</td>
<td>getHearingUser(): this method returns the name of a specific user requested by the administrator.</td>
</tr>
<tr>
<td>String getDeafUser()</td>
<td>getDeafUser(): returns the name of a specific Deaf user requested by the administrator.</td>
</tr>
<tr>
<td>Void setDeafuser()</td>
<td>setDeafuser(): changes the name of the Deaf user when requested by the administrator, from an old value to a new value or from a null to an existing value.</td>
</tr>
<tr>
<td>Void setHearinguser()</td>
<td>setHearinguser(): changes the name of the hearing user from the old value to a new one or from a null value to an existing one.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>String getTypePhone():</strong></td>
<td>returns the type of device the hearing user is using, whether hardphone or softphone</td>
</tr>
<tr>
<td><strong>Void setTypePhone():</strong></td>
<td>it changes the old value of the type of phone to a new value.</td>
</tr>
<tr>
<td><strong>Void addSoftphone():</strong></td>
<td>add the softphone device to the database.</td>
</tr>
<tr>
<td><strong>Void addHardphone():</strong></td>
<td>add the hardphone to the database which is used by the hearing user.</td>
</tr>
<tr>
<td><strong>Void viewDetails():</strong></td>
<td>it returns all SIP clients that are using the Asterisk server.</td>
</tr>
<tr>
<td><strong>Void editDetails():</strong></td>
<td>this method allows the administrator to edit details of all SIP clients i.e. if the URL of the user changes.</td>
</tr>
<tr>
<td><strong>Void removeDetails():</strong></td>
<td>this method allows the administrator to remove SIP clients in case they are not longer using the system.</td>
</tr>
<tr>
<td>AsteriskServer</td>
<td>Void registerSIPUsers(): this method register all SIP clients that have send requested service from the system.</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Void authenticateSIPUsers(): it validates all the SIP clients before they start communication.</td>
</tr>
<tr>
<td></td>
<td>Void deliverIM(): the method will route the instant message to the PJSIP client on PC.</td>
</tr>
<tr>
<td></td>
<td>Void deliverCall(): this method will route the call to the correct SIP client.</td>
</tr>
<tr>
<td></td>
<td>Void sendMSGSphinx(): this method will send the voice message to sphinx if the message needs speech to text transformation.</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Void sendMSGFestival()</td>
<td>This method will send the instant message to Festival server if the message needs text to speech transformation.</td>
</tr>
<tr>
<td>Void terminateCall()</td>
<td>This method terminates the call from the SIP client on a mobile phone.</td>
</tr>
<tr>
<td>Void connectToAsterisk()</td>
<td>This method establishes a connection to the Asterisk server.</td>
</tr>
<tr>
<td>Void makeCall()</td>
<td>Allows user to send voice calls to another SIP client registered in Asterisk server.</td>
</tr>
<tr>
<td>Void receiveCall()</td>
<td>Allows user to receive voice calls from other SIP clients.</td>
</tr>
<tr>
<td>Void addContacts()</td>
<td>Allows the user to add names of peers that the user would like to communicate with.</td>
</tr>
</tbody>
</table>

**Table 3: illustrates the methods and attributes**
State diagram

The state diagram illustrates the solution as moving from the starting state represented by the dot on the left and the final state represented by the dot on the right. The solution shows one way communication i.e. when the Deaf user on PJSIPClientOnPC(windows) sends an instant message to the hearing user on SIPClientOnMobilePhone.

Figure 8 shows the one way communication for the Deaf User

Pseudo-code

PJSIPClientOnPC(windows)
class PJSIPClientOnPC{
    int addressURL;
    int time;
    string Username;
    string password;
    int portno;
}
int message;

method sendIM()
{
    String to()
    String from()
    while(true)
    {
        send IM to PJSIP client on (ubuntu)
    }
}

method receiveIM()
{
    display instant message to PJSIP client;
}

method addContacts()
{
    getUserName()
    { return username of the PJSIP client; }
    getAddressURL()
    { return the IP address; }
}

method string to()
{ return URL address of the destination user; }

method string from()
{ return URL address of the sender; }

method string username()
{ return username of the PJSIPClientOnPc(windows); }

**Asterisk server**

class AsteriskServer{
    int extension;
    int username;
    int password;

    method registerSIPUsers()
    { return registration status; }
}
method authenticateSIPUsers() {
    return true if successfully registered;
    else{
        return false;
    }
}

method deliverVoiceCall() {
    method checkUserAvailability() {
        return status of user;
    }
}

method check>Type() {
    return either text or voice;

    if(message==voice) {
        method sendVoiceMSGtoSphinx()
    } else if (message==text) {
        method sendIMtoFestival()
    } else{
        return message not allowed;
    }
}

method EndCall() {
    return terminate connection to AsteriskServer;
}

Administrative Backend

method getHearingUser() {
    return all hearing users;
}

method getDeafUser() {
    return all deaf users;
}

method getTypePhone() {
    return softphone or hardphone;
}

method addSoftPhone() {
    return softphone added
}

method addHardPhone() {
    string name;
int extension;
string phoneDescription;
int portno;
return hardpphone added;
}
method ViewUserDetails()
{
    method getHearingUser()
    display all users;
    }
    method getDeafUserUser()
    display all users;
    }
    method editUserDetails()
    update user contact details;
    }
    method removeUserDetails()
    removes user contact details from the system;
    }

PJSIPClientOnPC(Ubuntu)

class PJSIPClientOnPC()
{
    method AcceptIMFromAsterisk()
    return InstantMessage;
    }
    method SendIMToAsterisk()
    if(message!=null)
    return direct message to Asterisk;
    }
    method ConnectToDatabase()
    getNameofDatabase()
    return name;
    }
    getPasswordDatabase()
    return password;
    }
    }
    method DisconnectFromDatabase()
    close database;
    }


IMPLEMENTATION

In this chapter the code itself is documented, which is a detail explanation of each line of the code on its contribution to the project. I further give a brief explanation of why speech-to-text with Sphinx was problematic and also the list of softwares that were installed during implementation. This honours project was also possible through the use of configuration files from Asterisk namely sip.conf and extensions.conf. These two files are the ones that I configured to establish connection between the instant messaging client (PJSIP) and the VoIP client(xlite) within Asterisk.

```python
#NAME OF FILE
#ReadMessage.py
#INPUT
#takes message strings
#OUTPUT
#writes the message strings to a another file
#CODE DESCRIPTION
#the code below takes every message that the Deaf client is sending to asterisk and writes in a separate file that is where festival reads the message, the message of the user is dropped in a file called messages which is one of asterisk's files and then i use this code to read from that file and write to a new file

#!/usr/bin/env python
def getLastLine():
    file = open('/var/log/asterisk/messages','r') #specify the file to open which is named messages in asterisk
    data = file.readlines() #read lines in file and put into file.close()
    lastline = data[len(data)-2] #-2 reads the second last item on the asterisk message file
```
lastline = lastline.replace("\n","")
return lastline

def printToFile(location,test):
    # this line prints the message to the file which is called test.txt
    file = open(location,'w')
    file.writelines(test)  # this line writes the message in the test file
    printToFile("/home/docas/Desktop/read_write/test.txt",getLastLine())
#NAME OF FILE(sphinx_agi.agi)
#INPUT
# takes audio files
# OUTPUT
# returns text
# CODE DESCRIPTION

# this file is called sphinx_agi.agi. This file is called every time an extension is
dialed, then the user will say a words "yes, no, accept and cancel"
# and the the audio files are then stored in a file,
# where they are upsample to 8khz so as it will be allowed through asterisk
# after upsampling the file is then passed to the asr engine sphinx
# using the sphinx client code sphinx-client.pl
# the result is only returned in
#!/usr/bin/perl –w

use Asterisk::AGI;
my $AGI = new Asterisk::AGI;
%input = $AGI->ReadParse();

sub asr {
  use IO::Socket;
  use FileHandle;
  use IPC::Open2;
  my $file = shift or return undef;
  my $host = shift || 'localhost';
  my $port = shift || '1069';
  my $fh;

  my $remote = IO::Socket::INET->new(Proto => "tcp",
      PeerAddr => "$host",
      PeerPort => "$port",
  ) or return undef;
  # Idea here being that you can pass a reference to an existing file handle
  if (ref $file) {
    my $fh = $file;
  } else {
    open (FH, $file) || return undef;  # opening the audio file and
    $fh = *FH;
  }
  # here i am only allowing audio files that are recorded in .gsm and .wav files
  $file =~ /(gsm|wav)/;  # comparing the file to .gsm and .wav
my $type = $1;
if ($type !~ /gsm|wav/) {
    warn "Unknown file type ($file)";
    return undef;
}

# the idea here is that i am using sinc interpolation (sox) to upsample the sample
rate to 8khz so that the audio files can be allow through asterisk
# sox is invoked with open2 and its told to upsample audio files to 8khz
$pid = open2(*SOXIN, *SOXOUT, "sox -t $type -s -r 8000 -w -t wav -2>/dev/null") || warn ("Could not open2.\n");

#binmode is used to write the files in binary format, since they are audio files
binmode $fh;
binmode SOXIN;
binmode SOXOUT;
binmode $remote;

# this while loop its reading from the audio file named my $file and pass it to sox
# for upsampling
while (defined(my $b = read $fh, my($buf), 4096)) {
    last if $b == 0;
    $count += $b;
    print SOXOUT $buf;
}
close SOXOUT;

# this following while loop reads the pieces of audio files that have
# been upsampled and store them in a variable called $buf

$count = 0;
my $sox = undef;
while (defined(my $b = read SOXIN, my($buf), 4096)) {
    last if $b == 0;
    $count += $b;
    $sox .= $buf;
}
print $remote length($sox) . "\n";
print $remote "$sox";}
close SOXIN;

$count=0;
while (defined(my $b = read $remote, my($buf), 4096)) {
    last if $b == 0;
    $count += $b;
    $result .= $buf;
}

close $fh;
close $remote;
return "$result";
}

# this subroutine is the one that passes the audio file that has been upsample by 
# sox and pass it to the asr engine(sphinx), this audio files are temporarily 
# stored in the /tmp/ folder where the asr engine retrieves them. then the audio 
# files are validated to check if the user only recorded "yes,no,accept,cancel) 
# otherwise the audio files are not converted to text.

sub confirm {
    while (my $tries <= 3) {
        $tries++;

        $AGI->stream_file("vr/say_yes_no","nnn");
        $AGI->stream_file("beep","nnn");
        $AGI->record_file("/tmp/$$", 'gsm', 0,3000);
        $AGI->stream_file("beep","nnn");

        #this is calling the asr sub from sphinx-netclient.pl
        #Again, this sub needs to be in this same script. $vresponse will contain the 
        #transcription of the what the caler said.
        my $vresponse = asr("/tmp/$$.gsm"); # it reads the recorde audio file 
        # from tmp directory and 
        # pass it to sphinx
        $AGI->verbose("CONFIRM: $vresponse");

        next if $vresponse !~ /YES|NO|ACCEPT|CANCEL/;
}
#the user is only allowed to either say "yes,accept or cancel"
#this are the only words i have prepared a language model for them

$gotresp = 1;

if ($vresponse =~ /NO|CANCEL/i) {
    sleep 1;
    $AGI->stream_file("cancelled",""");
    return undef;
} else {
    $AGI->set_variable('RESPONSE', 'YES');
    return 1;
}

if (! $gotresp) {
    sleep 1;
    $AGI->stream_file("invalid_selection",""");
    return undef;
}

$AGI->stream_file("vr/green_eggs_ham",""");
unless ( confirm() ) {
    #They said "no"
    $AGI->set_variable('RESPONSE', 'NO');
    exit;
}

;the below file is named extensions.conf from asterisk,it allows only the specified users to make calls and send messages to the server. This means that a user has a username and a corresponding extension next to his/her username

tutorial
exten => 4321,1,Dial(SIP/test)
exten => 8911,1,Dial(SIP/pulane)
exten => 5781,1,Dial(SIP/docas)
exten => 8888,1,Dial(SIP/thabile)
exten => 4545,1,Dial(SIP/111)
exten => 1234,1,Dial(SIP/ivan)
exten => 2005,1,Dial(SIP/dudu)
exten => 2222,1,Dial(SIP/helen)
exten => 3333,1,Dial(SIP/dave)
exten => 7777,1,Dial(SIP/charles)
exten => 1111,1,Dial(SIP/charl)
;the lines below are for testing if i can call perl scripts in the asterisk directory called agi-bin
exten => 257,1,Answer()
exten => 257,2,AGI(agi-test.agi)
exten => 257,3,Hangup

;
exten => 1212,1,Answer()
exten => 1212,2,AGI(sphinx-agi.agi)
exten => 1212,3,hangup

;below are the lines that call the python code read.py that takes the code from the command line of asterisk and place it nicely in a text file
exen => 1987,1,wait(1)
exen => 1987,2,System(/usr/bin/python /home/docas/Desktop/mess1.py) ;the system command i used to run the perl script that has the message from the user
exen => 1987,3,Hangup

;the lines below is where festival is fed with text from the file,festival reads what’s in the text file when the extensions 667 is dialled
exen =>
667,1,ReadFile(test=/home/docas/Desktop/read_write/test.txt,130)
exen => 1667,2,Festival(${test})
exen => 667,3,Hangup

Difficulties encountered

Asterisk's lack of support to Instant Messaging
Asterisk does not allow messages out of a call ,this means that messages are only allowed within a call of which those are call notifications. According to the information got from asterisk developers from Digium through Asterisk user’s forum and also working with Asterisk for the past six months a conclusion can be made that Asterisk does not support instant messaging.

Well a solution to the above is to send the message straight to the Asterisk server A python script is then used to extract the message from the server and writes it
in a separate file where festival reads it and Asterisk routes it to the hearing client, but the best way is to use either a server that supports VoIP and IM or peer-to-peer.

**Sphinx only support 16khz sample rate**

“It is fairly easy to integrate Asterisk with Sphinx; the only trouble is that you need to have an Acoustic Model(AM) for 8kHz which are not (yet) readily available.

There is a Language Model(LM) and AM included with Sphinx, but it is designed for a sampling rate of 16kHz and therefore does not work with Asterisk. Even if you use sophisticated upsampling techniques(sinc interpolation-sox) to create 16kHz for use with Sphinx, the recognition rate is absolutely dismal” (Eldeman, 2006)

The sample rate on Sphinx with Asterisk is the major problem on speech-to-test, Asterisk expects small phrases of 8khz and 8khz.wav or .gsm phrases does not produce good sound quality and Sphinx fails to perform recognition if the sound quality is not good. Therefore could not get Sphinx to work with Asterisk. The Sphinx mentioned above is Sphinx version 2, however there is version 3 & 4 already but this versions has been improved on the recognition rate meaning that the sample rate is even higher than 16kHz. Therefore many people on google have not successfully configured Asterisk with Sphinx and all these people are following one tutorial, even the questions posed by the viewers were not answered by the author. Therefore there is no much information on google concerning Asterisk integration with Sphinx.

Solution to this is to try do the speech-to-text using Sphinx peer-to-peer without Asterisk within the remain month of honours.

To conclude Asterisk is a very easy and understandable server and works best for VoIP but did not play a major role on this project.
### Softwares installed

<table>
<thead>
<tr>
<th>Tools</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Festival 1.96 -beta</td>
<td>Multi-lingual software speech synthesizer</td>
</tr>
<tr>
<td>Festival-cmu</td>
<td>CMU dictionary for festival</td>
</tr>
<tr>
<td>Festvox-kallpc-16k</td>
<td>American english male speaker</td>
</tr>
<tr>
<td>Sphinx2.0.6</td>
<td>Automatic speech recognition engine</td>
</tr>
<tr>
<td>Libsphinx2-dev</td>
<td>Speech recognition library-development kit</td>
</tr>
<tr>
<td>Sphinx2-hmm-6k</td>
<td>Speech recognition library-default acoustic model</td>
</tr>
<tr>
<td>Asterisk 1.4.17</td>
<td>VoIP server</td>
</tr>
<tr>
<td>Microsoft visual studio 2005</td>
<td>IDE</td>
</tr>
<tr>
<td>Microsoft SDK</td>
<td>libraries</td>
</tr>
<tr>
<td>Xlite softphone 3.0</td>
<td>Voip client</td>
</tr>
<tr>
<td>PJ SIP</td>
<td>IM client</td>
</tr>
</tbody>
</table>
Chapter 8

TESTING

The previous chapter focused on the implementation itself. It gave a detailed documentation of the code used and explained how each part has contributed on the project. The testing chapter will discuss the usability, functionality and the performance of the system and then evaluate the results. The process of testing is documented for both the user interface and the system, to ensure that the solution meets the system requirements and that it is robust.

The types of testing that will be conducted are usability testing, functionality testing and the performance; the basic requirement for all three kind of testing is to firstly identify the target group, recruit the users, establish the task, carry out the evaluation and report on the findings.

The testing will be conducted in the lab with four participants, two users Deaf and two users hearing; two of these candidates can also be used as administrators. All users were given the appropriate user manual that they should follow (see appendices). Two candidates will be given a scenario that they need to perform using the system.

Preparing for the test requires the following:

- 4 candidates will be needed 2 Deaf and 2 hearing
- 2 PCs (1 PC for hearing client 1 PC for Deaf client)
- Xlite installation (for both hearing client and Deaf Client)
- Asterisk installation (for the administrator)
Tests

- The friendliness of the system: the users will not be told how the system operates
- The time complexity of the system: how long does it take the mother to wait for a reply?
- Most importantly how comfortable are the users with the voice that the system is using?

Experimental procedure

Firstly all users were given the user guides; the Deaf user was given the user manual for Deaf user and hearing user (see appendix D) and the administrator was given a user manual for the administrator (refer to Appendix D) to read and follow and then it was explained to them that they should follow a scenario that is given to them; during this process the functionality, the performance and the usability of the system was evaluated. Now for the two users to start communicating the Deaf user clicks to one of the saved contacts, type the message and click send and the hearing user dials a number received from the administrator to retrieve the message. After completing the task the users were given the system questionnaires (refer to Appendix E) to complete as well as verbal feedback.
Data collection

Scenario

Suppose a hearing mother is teaching her Deaf daughter how to do basic mathematics. The mother will pose a question that the daughter must differentiate a basic equation by sending a text message. The daughter receives the equation that she is supposed to do, however the daughter will have questions along the way whenever she is stuck. Since the daughter is Deaf she will send an IM message to the mother asking questions until she eventually gets it right.

Description of how the scenario was conducted

The daughter was given a simple equation 4x+4; 9x+9; x^2+4x+4 the mother will explain the procedure of how to solve the equation e.g. the mother will type: firstly open four brackets and then find two numbers such that when you multiply them you get the last term and when you add them you get the middle term.

The following is a brief description of the four participants that were used during testing. Based on the results of the demographic questionnaires 50% of the users were females and 50% were males; 4 out of 4 users have used a computer before therefore they are good participants for the testing of the project because it requires a participant to know how to type a message and send it.

<table>
<thead>
<tr>
<th>Participant</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age:</td>
<td>20</td>
<td>19</td>
<td>25</td>
<td>23</td>
</tr>
<tr>
<td>Gender:</td>
<td>Female</td>
<td>Female</td>
<td>Male</td>
<td>Male</td>
</tr>
<tr>
<td>Computer literate:</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>Yes</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
</tr>
<tr>
<td>Highest qualification:</td>
<td>Undergrad. degree</td>
<td>Current Undergrad</td>
<td>Current Post grad</td>
<td>Current Post grad</td>
</tr>
</tbody>
</table>

*Table 3: participant’s details*
**Test results**

**Usability testing**

According to results obtained from the background questionnaires 4 out 4 users excluding users who acted as administrators have access to Instant Messaging clients and also have access to the internet. Therefore users found it very easy to use the X-lite softphone as an Instant Messaging client for the user who is acting as a Deaf user and also using X-lite VoIP for the user who is acting as a hearing user. As for the administrator following the user manual, the administrator found it very easy to find the location of all the required files and also modifying those files was not difficult at all. There was no mismatch of users when adding users to the sip.conf and extention.conf since each user is identified by its name.

**Functionality testing**

The users felt that the system met the necessary requirement of text message convention because the system was able to convert all messages that were forwarded by the Deaf user. There was no delay of text messages sent to the Festival engine, messages were converted one after the other i.e. in the order that they were received by Festival engine.

**Performance testing**

The system was also tested on the time it takes for the process of sending the message to Festival and then the hearing user retrieving it. A stopwatch was used to find out the elapsed time. The time it took was approximately 9 seconds which is slow compared to two users chatting using an Instant Messenger, but since this system is an automated voice relay it is very fast compared to a system that does TTS using a human operator.

The system was also tested when multiple users are retrieving their messages at the same time, the system responded very good because there was no slowing down of the system, the messages were converted one after the other but when
the two users dialed their own assigned numbers they were able to listen to their messages simultaneously. Most users agreed that the system does meet the necessary requirements; the following graph illustrates how the users voted on the overall performance of the system.

![Figure 4: performance testing](image)

*Figure 4: performance testing*
**Areas of improvement**

2 out of 4 hearing users said they would have loved that if a female Deaf user is texting then a female voice should be used and also if a male Deaf user is texting then a male voice be used. Hearing users felt that the accent of the voice is all not good, since the system is using a British male voice that it would have been better if the system will preferably use a South African accent it would have been much better though it was explained to them that the TTS system does not yet have a South African accent ready.


A p p e n d i x  A

INSTALLATION GUIDE FOR THE ADMINISTRATOR

Asterisk
The administrator has to manage, upgrade and maintain Asterisk for the benefit of the clients. The following is a list of steps to be followed by the administrator. Firstly you need to download Asterisk from http://cs.uwc.ac.za/~zdudu/sourcecode/asterisk.tar.gz and then the following packages are needed before compiling Asterisk (libncurses,libncurses-dev). The operating system used for the project is Linux; therefore Asterisk has to be installed using the following steps on linux.

Step 1: open up your terminal and untar asterisk.tar.gz as tar zxf asterisk.tar.gz under any folder name you like.

Step 2: Compile Asterisk by typing ./configure in your terminal, successful configuration should look like the following

Step 3: make ; make install

Step 4: start the Asterisk server as follows asterisk and then asterisk -r to connect
Festival

Festival is also needed to provide the text-to-speech capability. To install Festival the following command applies

**Step 1:** to install type `sudo aptitude install festival` on the terminal.

**Step 2:** you need a voice to say the text, to install the voice first type `sudo aptitude search festvox` to select a voice that you would like to use and then `sudo aptitude install` selected voice.

**Step 3:** to start the server type `festival --server`
The administrator will register users and assign extensions for respective registered users in Asterisk. To register users the following steps should be used

**Step 1:** `sudo -i` to login as root and type password

**Step 2:** `vim /etc/asterisk/sip.conf` the following snapshot is an example of how to add both Deaf and hearing users to sip.conf

```
[ivan]
Username=ivan
Authname=ivan
Secret=pwdivan
Type=friend
Host=dynamic
Context=tutorial

[sinle]
Username=sinle
Authname=sinle
Secret=pwdsinle
Type=friend
Host=dynamic
Context=tutorial
```

**Step 3:** Now to connect the messages from the Deaf user to the Festival server, you need to assign extensions to all hearing users; it is strongly recommend that you add at the end of extensions.conf file unless you know what you are doing.

To go to the file go to `vim/etc/asterisk/extensions.conf`

```
e.g. exten => 667,1, ReadFile(test=/home/docas/text.txt,130)
    exten =>667,2,Festival($test})
    exten => 667,3,hangup
```

**NB:** an empty text file need to be created and the location of it need to be specified in the ReadFile and also Festival server need to be running.
The following is a snapshot of festival server when text message are being send to the server i.e. when 667 is dialed, note that 667 is an example you can use any desired number.

```
docas@docas-desktop:$ sudo -i
[sudo] password for docas:
root@docas-desktop:~$ festival --server
server  Mon Oct 19 01:12:48 2009 : Festival server started on port 1314
client(1) Mon Oct 19 01:17:45 2009 : accepted from localhost
client(1) Mon Oct 19 01:17:45 2009 : disconnected
client(2) Mon Oct 19 01:17:58 2009 : accepted from localhost
client(2) Mon Oct 19 01:17:58 2009 : disconnected
```

In the above example a hearing user will dial 667 to retrieve the voice message, note that all users need to be registered in sip.conf and extensions.conf in Asterisk server before they can use the server.
Appendix C

INSTALLATION GUIDE FOR HEARING AND DEAF USERS

Installing Xlite

The hearing user will need to have an xlite softphone or any sip softphone. To download xlite follow this link http://www.counterpath.com/x-lite.html. The Deaf user has a choice of either using PJSIP as an Instant Messaging client or X-lite, however it is recommended that X-lite is used as an Instant Messaging client instead unless a user is familiar with programming environment. To download and install PJSIP follow this link http://www.pjsip.org. To download xlite follow this link http://www.counterpath.com/x-lite.html, then follow these steps.

Run the X-Lite setup executable file and follow the prompts from the install wizard.
At the final step of the wizard, check the Launch X-lite checkbox to start using the softphone
Click Finish to complete the installation guide

The picture shows X-lite after successful installation
To use X-lite you need to first configure it.

**Configuring X-lite**

Start X-lite by using the Windows Start menu or by double-clicking the desktop icon. The Call display shows Initializing, followed by Discovering network and Awaiting proxy login information. Users must set up at least one account before using X-lite to place or receive calls. For information about account setup, see “Setting up Accounts” below.

To use an account for VoIP communication, the account must be enabled within the X-Lite client. To enable an account, click at the top of the softphone, choose **SIP Account Settings** and select the **Enable** checkbox for the desired account. The following is a picture of what you should see note that to the username, the passwords and the domain should be obtained from the administrator.
Figure 9: Xlite settings
On screen Display

For the hearing user

To dial click the phone number given by the administrator and then press the green dial button as show from the on screen display.

For the Deaf user

To send an instant message:
On the calls& contacts as shown on the on screen display above click any person you would like to send an instant message to then a window that is the same as the one below should appear, type your message then click the send button.
Figure 10: Xlite Instant Messaging
Appendix E

Questionnaires

Background questionnaires

How proficient are you in English?


How often do you use an instant messenger?


Do you have access to a computer?

Yes No

How comfortable are you in accessing the internet on a computer?


System testing questionnaires

Is the product easy to use?


Is the product user friendly?

What difficulties have you encountered when using the product? If yes explain
________________________________________________________________________
________________________________________________________________________
________________________________________________________________________

What is the overall performance of the product? 
4. Very good  3.good  2.acceptable  1.poor

4 3 2 1

What improvements would you suggest if any?
________________________________________________________________________

Does it meet the necessary requirements? 
4. very often  3.often  2. Rarely  1. never

4 3 2 1
## Project plan

<table>
<thead>
<tr>
<th>Term 1</th>
<th>Term 2</th>
<th>Term 3</th>
<th>Term 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read articles on Asterisk; found out more about Asterisk and its capability</td>
<td>Updated thesis document (RAD); wrote OOA, UIS and OOD</td>
<td>Implemented text-to-speech with Festival</td>
<td>Sphinx integration with Asterisk (not done)</td>
</tr>
<tr>
<td>Wrote URD, RAD</td>
<td>Implemented text-to-speech prototype</td>
<td>Setup IM proxy with Jabber server, installed and configured wildfire as a jabber server</td>
<td>Text-to-speech testing and improvement</td>
</tr>
<tr>
<td>Installed Asterisk with its depended packages; installed xlite and get it do basic voice calls within Asterisk</td>
<td>Setup an Asterisk-IM plug-in with wildfire server</td>
<td>User manual and installation guide for administrator, Deaf and hearing user</td>
<td></td>
</tr>
<tr>
<td>Website designing; installed Festival and Sphinx</td>
<td>Trained sphinx with simple phrases as a standalone</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>